Graph Pricing Problem on Bounded Treewidth, Bounded Genus and k-partite graphs

Parinya Chalermsook* Shiva Kintali† Richard Lipton‡ Danupon Nanongkai§

Abstract

Consider the following problem. A seller has infinite copies of n products represented by nodes in a graph. There are m consumers, each has a budget and wants to buy two products. Consumers are represented by weighted edges. Given the prices of products, each consumer will buy both products she wants, at the given price, if she can afford to. Our objective is to help the seller price the products to maximize her profit.

This problem is called *graph vertex pricing* (GVP) problem and has resisted several recent attempts despite its current simple solution. This motivates the study of this problem on special classes of graphs. In this paper, we study this problem on a large class of graphs such as graphs with bounded treewidth, bounded genus and k-partite graphs.

We show that there exists an FPTAS for GVP on graphs with bounded treewidth. This result is also extended to an FPTAS for the more general *single-minded pricing* problem. On bounded genus graphs we present a PTAS and show that GVP is NP-hard even on planar graphs.

We study the integrality gap of the Sherali-Adams relaxation of the natural LP which has gained much interest recently as a possible approach to develop new approximation algorithms. We show that, when the input graph has bounded treewidth or bounded genus, applying constant number of rounds of Sherali-Adams relaxation makes the integrality gap of the natural LP arbitrarily close to one.

On k-partite graphs, we present a constant-factor approximation algorithm. We further improve the approximation factors for paths, cycles and graphs with degree at most three.

*Department of Computer Science, University of Chicago, Chicago, IL. Email : *parinya@cs.uchicago.edu*

†Department of Computer Science, Princeton University, Princeton, NJ-08540. Email : *kintali@cs.princeton.edu*

‡College of Computing, Georgia Institute of Technology, Atlanta, GA-30332. Email : *rjl@cc.gatech.edu*

§Theory and Applications of Algorithms Research Group, University of Vienna, Vienna, Austria. Email : *danupon@gmail.com*. Work partially done while at Georgia Institute of Technology, Georgia, USA.

1 Introduction

Consider the following problem where a seller is trying to sell her products to make the most profit. The seller has infinite copies of n products represented by nodes in a graph $G(V, E)$. She knows that there are m consumers who want to buy exactly two products each. Each consumer is represented by an edge e in G and has a budget B_e. She will buy both products (represented by the end vertices of the edge e) if the price of both products together does not exceed her budget B_e. The seller's goal is to price all products to make the most revenue. That is, she wants to find a price function $p : V(G) \to \mathbb{R}_+ \cup \{0\}$ that maximizes

$$\sum_{uv \in E(G)} \begin{cases} p(u) + p(v) & \text{if } p(u) + p(v) \leq B_e, \\ 0 & \text{otherwise.} \end{cases}$$

This problem is called the *graph vertex pricing* (GVP) problem. It is one of the fundamental special cases of the *single-minded item pricing* (SMP) problem. In SMP, consumers may want to buy more than two products (thus we can represent the input by a hypergraph with budgets on the hyperedges). This problem arises from the application of pricing digital goods (see [GHK+05] for more details).

Both GVP and SMP are proposed by Guruswami et al. [GHK+05] along with an $O(\log n + \log m)$ approximation algorithm for the SMP problem and an APX-hardness of the GVP problem. Balcan and Blum [BB07] presented a surprisingly simple algorithm achieving a factor four approximation for the GVP problem. The hardness of approximation of GVP was recently shown to be $2 - \epsilon$, assuming the Unique Games Conjecture [KKMS09].

The algorithm of Balcan and Blum first constructs a bipartite graph by randomly partitioning the vertices into two sides and deleting edges connecting vertices on the same side. It then picks one side randomly and prices all vertices on that side to zero. The resulting instance can be solved optimally. The approximation guarantee of four follows from the fact that each edge is deleted with probability half and pricing vertices on one random side to zero reduces the optimal revenue by half (in expectation). The algorithm can be derandomized using standard techniques.

Surprisingly, this simple algorithm has stood against several attempts to improve the approximation factor. It is thus natural to attack the problem by first studying some special cases. In general, it is interesting to explore how the combinatorial structure of the input graph influences the approximability of the problem. Understanding special cases might lead to improving the upper bound for the general case, as understanding the case of bipartite graph lead to a 4-approximation algorithm for the general case. This line of attack was initiated recently in [KKMS09] where it is shown that the GVP problem is APX-hard on bipartite graphs, and in [KMR11] where an improved algorithm is presented for the case where the range of consumers' budgets is restricted.

In this paper, we continue this line of study and present approximation algorithms and hardness results of the GVP problem on many classes of graphs such as bounded treewidth graphs, bounded genus graphs and k-partite graphs.

1.1 Our Results

Bounded treewidth graphs and hypergraphs: To understand the structure of GVP, trees are a natural class of graphs to study. GVP is not known to be NP-hard on trees. We present an FPTAS for trees based on a dynamic programming algorithm. We generalize our algorithm to

bounded treewidth graphs by applying our technique on the tree decomposition of the input graph, which can be computed in linear time by an algorithm of Bodlaender [Bod96]. We extend our algorithm to give an FPTAS for bounded treewidth *hypergraphs* as well.

Bounded genus graphs: Bounded genus graphs are broad class of graphs which play a major role both in structural graph theory and algorithmic graph theory. Several important NP-hard optimization problems admit improved algorithms on planar graphs (i.e., graphs with genus zero). We present a PTAS for GVP on bounded genus graphs and show that it is NP-hard even on planar graphs.

Our result relies on Baker's and Eppstein's Techniques [Bak94, Epp00]. However, while such previous results presented a polynomial time exact algorithms on graphs with bounded treewidth, it does not seem to be the case in the GVP problem. Instead, we show that the GVP admits fully polynomial-time approximation scheme (FPTAS) on bounded treewidth graphs.

To the best of our knowledge, the only other work that studied any variation of the pricing problem on graphs with bounded treewidth or bounded genus is [CDF+09], where it is shown that the *Stackelberg minimum spanning tree pricing* (SMST) problem can be solved in polynomial time on bounded treewidth graphs and NP-hard on planar graphs. The question whether there is a PTAS for SMST on planar graphs is still open. Thus, our result is the first PTAS for a pricing problem on planar graphs and it extends to bounded genus graphs.

Integrality gap of the Sherali-Adams relaxation: We study the integrality gap of the linear program in the Sherali-Adams hierarchy [SA90]. Sherali-Adams relaxation is one of the lift-and-project schemes which have received much interest recently in the approximation algorithms community (see, e.g [KMN10] and references therein). A question of particular interest is how the integrality gaps evolve through a series of rounds of Sherali-Adams relaxation. Positive results (i.e., small integrality gap) could potentially lead to an improved algorithm while negative results rule out a wide class of approximation algorithms.

In this paper, we show a positive result that the integrality gap is $(1+\epsilon)$, for any ϵ, after $O_\epsilon(\min(g,w))$ rounds of Sherali-Adams relaxation where g and w are the genus and the treewidth of the input graph respectively and the constant in O_ϵ depends on ϵ. To our knowledge, besides our work, the only work that studies the Sherali-Adams relaxation in the realm of pricing problem is [KKMS09] where a lower bound of $4-\epsilon$ on the integrality gap is shown. It can be observed that the LP of [KKMS09] is equivalent to the result of two rounds of Sherali-Adams lift-and-project operation on our LP. Our work is the first result that studies the integrality gap of this problem when *many* rounds of Sherali-Adams relaxation are allowed and suggests that Sherali-Adams relaxation might be useful in attacking the GVP problem on general graphs.

Sherali-Adams relaxation for combinatorial optimization problems on bounded treewidth graphs has been considered before, most notably in [BÖ04, CKR10]. Our rounding algorithm (although discovered independently) is similar to the algorithm in [BÖ04]. We refer the readers to these papers for a more complete survey on Sherali-Adams relaxation.

k-partite graphs: We also study the GVP problem on k-partite graphs. This class of graphs generalizes the class of bipartite graphs used to develop the 4-approximation algorithm of [BB07] and includes graphs of bounded degree (due to Brooks' theorem [Bro41]) and graphs of bounded genus (due to Heawood's result [Hea90]).

Since the GVP problem is APX-hard even on bipartite graphs [KKMS09], we cannot hope for a PTAS for k-partite graphs. We present a $(4 \cdot \frac{k-1}{k})$-approximation algorithm when k is even and a $(4 \cdot \frac{k}{k+1})$-approximation when k is odd. This gives a slight improvement on the approximation factor on general graphs. We show that improving these bounds further, when k is even, is as hard as improving the 4-approximation factor for the general graphs.

Bounded Degree graphs: Finally, we study the problem on graphs of degree at most two and three. For graphs of degree at most two (i.e., paths and cycles), we show that the GVP problem can be solved optimally in polynomial time. For graphs of degree at most three, the previous result on tri-partite graphs implies a 3-approximation algorithm since these graphs are 3-colorable (by Brook's theorem [Bro41]). We present a different algorithm that is a 2-approximation for this class of graphs.

Organization: In Section 3, we present algorithms for graphs of bounded tree-width. We show NP-hardness proof for GVP on planar graphs in Section 4, and provide a polynomial time approximation scheme. We discuss the integrality gap of Sherali-Adams relaxation in Section 5. We present algorithms for k-partite graphs and graphs of bounded degrees in Section 6.

2 Preliminaries

Integral and polynomially bounded budget assumption: We argue that we may assume w.l.o.g. that optimal prices, as well as budgets, are integral and polynomially bounded. The proof of this fact uses standard techniques and is deferred to Appendix A.

Lemma 2.1. Let $(G, \{B_e\}_{e \in E(G)})$ be an input instance. Then for any $\epsilon > 0$, we can find, in polynomial time, another set of budgets $\{B'_e\}_{e \in E}$ such that

- For all $e \in E$, B'_e is integral and has value at most $O(mn/\epsilon)$.
- There is a price p' such that $p'(v)$ is integral for all v, and the revenue of p' is at least $(1-\epsilon)\mathsf{OPT}'$, where OPT' is the optimal solution of the new instance.
- Any γ approximation algorithm for an instance $(G, \{B'_e\})$ can be turned into $(1+\epsilon)\gamma$ approximation algorithm for the original instance.

Treewidth : Let G be a graph with treewidth at most k. By definition (see, e.g., [RS84]), there is a *tree-decomposition* $(T, \mathcal{V})$ of G of width k; that is, there is a tree T and $\mathcal{V} = (V_t)_{t \in T}$, a family of vertex sets $V_t \subseteq V(G)$ indexed by the vertices t of T, with the following properties.

1. $V(G) = \bigcup_{t \in T} V_t$;
2. for every edge $e = uv \in G$ there exists $t \in T$ such that both u and v lie in V_t;
3. for any $v \in V(G)$, if $v \in V_{t_1}$ and $v \in V_{t_2}$ then $v \in V_{t_3}$ for any t_3 in the (unique) path between t_1 and t_2;
4. $\max_{t \in T} |V_t| = k + 1$.

For any fixed constant k, Bodlaender [Bod96] presented a linear time algorithm that determines if the treewidth of G is at most k and if so constructs a corresponding tree decomposition.

3 FPTAS on bounded treewidth graphs

We denote by $P = O(mn/\epsilon)$ the maximum possible budget and prices, as obtained from Lemma 2.1.

Let G be any input graph with edge weights satisfying Lemma 2.1. Now, assuming that we have the tree decomposition $(T, \mathcal{V})$, we solve the problem in time poly$(|T|, P^k)$, where P is as in Lemma 2.1, as follows.

First, in addition to V_t for $t \in T$, we define set E_t as follows. Initially, we let $E_t = \emptyset$. For each edge $e \in E(G)$, let $t(e)$ be the vertex in T such that both end vertices of e are in V_t that is nearest to the root. Note that t exists by the second property of tree decomposition (cf. Section 2) and is unique by the third property. We add each edge e to $E_{t(e)}$. For any $t \in T$, let T_t be the subtree of T rooted at t.

For any vertex t with $V_t = \{v_1, \ldots, v_{k+1}\}$ and integers $Q_1, \ldots, Q_{k+1} \in [P]$, we have table entry $R[t, Q_1, \ldots, Q_{k+1}]$ defined to be the maximum revenue we can get from edges in $\bigcup_{t' \in T_t} E_{t'}$ when the price of v_j is Q_j for all $j = 1, \ldots, k+1$. Note that the size of table R is poly$(|T|, P^k)$.

We compute $R[t, Q_1, \ldots, Q_{k+1}]$ as follows. When t is a leaf node, we set the price of v_j to be Q_j for all j. Note that for all $e \in E_t$, both end vertices of e are in V_t. Thus, we can compute the revenue we obtain from edges in E_t without pricing any vertices outside V_t and so we can compute $R[t, Q_1, \ldots, Q_{k+1}]$. For non-leaf node t, we use the following equality.

$$R[t, Q_1, \ldots, Q_{k+1}] = \sum_{t' \in C(t)} \max_{Q'_1, \ldots, Q'_{k+1}} r(t', Q'_1, \ldots, Q'_{k+1}) + f(t, Q_1, \ldots, Q_{k+1})$$

where we define $r(t', Q'_1, \ldots, Q'_{k+1})$ and $f(t, Q_1, \ldots, Q_{k+1})$ as follows. We let $r(t', Q'_1, \ldots, Q'_{k+1})$ equals $-\infty$ if there exists j' and j such that $v_{j'} \in V_{t'}$ and $v_j \in V_t$ such that $v_{j'} = v_j$ and $i'_{j'} \neq i_j$ (in other words, the same node is set to price i_j at vertex t but to different price $i'_{j'}$ at vertex t'). Otherwise, $r(t', Q'_1, \ldots, Q'_{k+1}) = R[t', Q'_1, \ldots, Q'_{k+1}]$. Intuitively, we use $r(t', Q'_1, \ldots, Q'_{k+1})$ to make sure that every vertex receives only one price.

We define $f(t, Q_1, \ldots, Q_{k+1})$ to be the revenue we receive from edges in E_t when we price v_i to Q_i, for all i. Note that we can compute $f(t, Q_1, \ldots, Q_{k+1})$ without pricing vertices outside V_t since, by definition, both end vertices of every edge in E_t are in V_t.

Remark: We note that this algorithm could be easily modified to give FPTAS for bounded treewidth hypergraphs (as defined in [RS84]) as well. See Appendix B for more details.

4 Graphs of Bounded Genus

In this section we study graphs of bounded genus g. First we show that the problem is strongly NP-hard even for planar graphs ($g = 0$). This implies that there is no FPTAS for this special case. Then, we design a PTAS for it. This settles the complexity for the bounded genus case.

4.1 Hardness

Theorem 4.1. The graph vertex pricing problem on planar graphs is strongly NP-hard.

Proof. We show a reduction from Vertex Cover on planar graphs, which was shown to be strongly NP-hard in [GJ77]. Assuming that we are given the instance $G = (V, E)$ of vertex cover, we

construct the instance G' of GVP as follows. We add a new vertex v' for each $v \in V$, and v' is only connected to v but not to any other vertices; it is possible to do so without violating planarity. For each edge $e \in E$, we make a copy of e and add it to the instance. Call the resulting instance $G' = (V \cup V', E \cup E')$, where $|V'| = |V|$ and $|E'| = |V| + |E|$. We will have two types of consumers: the *rich consumers* that correspond to (parallel) edges of E have budget $|V|^2$ and $2|V|^2$ respectively, and the *poor consumers* corresponding to newly added edges of the form vv' have budget of 1.

The intuition for this construction is that the original edges have much more budget, so optimal solution would not try to miss any of those edges. And therefore the optimal solution would choose the vertex cover of E. Now we proceed to the analysis.

We let VC denote the size of minimum vertex cover of original instance G, and OPT denote the optimal revenue of the pricing problem on the instance G' constructed from G using the above reduction. We claim that $\mathsf{OPT} = 2|E||V|^2 + |V| - \mathsf{VC}$, and once we prove this claim, we would be done. We first show that $\mathsf{OPT} \geq 2|E||V|^2 + |V| - \mathsf{VC}$. Let $S \subseteq V$ be the set of vertices in an optimal vertex cover of G. We define the following price p: (i) set the price $p(v) = |V|^2$ and $p(v') = 0$ for each vertex $v \in S$, and (ii) $p(v) = 0$ and $p(v') = 1$ for each $v \notin S$. Notice that we can collect $2|V|^2$ from each rich consumer, resulting in a total of $2|E||V|^2$, while we can collect $|V| - \mathsf{VC}$ from the poor consumers.

For the converse, let p be an optimal price for G' and $C = \{v \in V : p(v) > 1\}$, i.e. C is the set of vertices v whose corresponding poor consumers vv' do not have enough money.

Claim 4.2. Set C forms a vertex cover of G.

Proof. Suppose not. Then there is an edge $uv \in E$ not covered by any vertices in C. Therefore, the revenue of uv is at most $p(u) + p(v) \leq 2$. The total revenue from this pricing is at most $2(|E|-1)|V|^2 + |V| + 4$, which is at most $2|E||V|^2 - |V|^2$, contradicting the fact that p is an optimal price since we have already proved that $\mathsf{OPT} \geq 2|E||V|^2$. □

So we know that $|C| \geq \mathsf{VC}$, and thus the total revenue from poor consumers is at most $|V| - \mathsf{VC}$. This implies that $\mathsf{OPT} \leq 2|E||V|^2 + |V| - \mathsf{VC}$, concluding Theorem 4.1 □

4.2 Algorithm

In this section, we describe a PTAS for GVP on graphs of bounded genus. Let g be the genus of the graph. It is known that any graphs of bounded genus are minor-closed, and therefore there are a constant number of forbidden minors. We will be using the following theorem, due to Demaine et.al.

Theorem 4.3. (Theorem 3.1 in [DHiK05]) For a fixed H, there is a constant c_H such that, for any integer $k \geq 1$, and for every H-minor free graph G, the vertices of G can be partitioned into $k+1$ sets such that any k of the sets induce a graph of treewidth at most $c_H k$. Furthermore, such a partition can be found in polynomial time.

We choose the parameter $k = \lceil 4/\epsilon \rceil$ and invoke the theorem to partition the vertex set $V(G)$ into $V(G) = \bigcup_{i=1}^{k} V_i$. We create k instances $H_1, \ldots, H_k$ where instance H_j is obtained by removing vertices in V_j and their adjacent edges from G. From the theorem, each graph H_j has treewidth at most $O(1/\epsilon)$.

Claim 4.4. $\sum_{j=1}^{k} \mathsf{OPT}(H_j) \geq (k-2)\mathsf{OPT}$ where $\mathsf{OPT}(H_j)$ denotes the optimal value for graph H_j.

Proof. Let p^* be an optimal price for graph G that collects the revenue of OPT. Let $E^* \subseteq E(G)$ be the subset of edges e such that $p^*(e) \leq B_e$. These are the edges that contribute to the revenue OPT. For each subset of vertices $S \subseteq V$, denote by $\delta_{E^*}(S)$ the set of edges in E^* with exactly one endpoint in S, and $E^*(S)$ the set of edges in E^* with both endpoints in S.

For each graph H_j, if we set the price p^*, we would be able to collect the revenue of at least $r_j = \sum_{i \neq j} \sum_{e \in E^*(V_i)} p^*(e) + \sum_{e \in \tilde{E} \setminus \delta_{E^*}(V_i)} p^*(e)$, where $\tilde{E}$ denotes the set of edges connecting two vertices in different sets V_i. Summing over all j, notice that each edge in $E^*(V_j)$ contributes exactly $k-1$ times for each j, while each edge in $\delta_{E^*}(V_j)$ contributes exactly $k-2$ times (edge uv connecting V_i to $V_{i'}$ contributes to all terms except for r_i and $r_{i'}$). This implies that $\sum_{j=1}^{k} r_j \geq (k-2)\mathsf{OPT}$. □

Let j^* be the index such that $\mathsf{OPT}(H_{j^*})$ is maximized, so we have $\mathsf{OPT}(H_{j^*}) \geq (1-\epsilon)\mathsf{OPT}$. Since the treewidth of H_{j^*} is at most $O(1/\epsilon)$, we can compute the optimal pricing in H_{j^*} in time $n^{O(1/\epsilon)}$.

5 Integrality gap of the Sherali-Adams relaxation on bounded treewidth and bounded genus graphs

We describe a family of LP relaxation, denoted by (LP-r), and a rounding algorithm that computes an optimal solution given an optimal fractional solution for (LP-r), provided that the treewidth of the input graph is at most $r/2$. One can use a standard procedure to check that the constraints of (LP-r) can be generated by applying $O(r)$ rounds of Sherali-Adams relaxation. For completeness, we provide the proof in Appendix C.

Terms and notation: Let P be the maximum possible price obtained from Lemma 2.1. Given an input graph $G = (V, E)$, let $p, p' : V \to [P]$ (or equivalently $p, p' \in [P]^V$) be price functions that assign prices to vertices in graph G. We say that two functions p, p' *agree* on S if and only if $p(v) = p'(v)$ for all $v \in S$. For any subset of vertices $S \subseteq V$, a *restriction* of p on set S, denoted by $p|_S$, is the (unique) function $\tilde{p} \in [P]^S$ that agrees with p on S.

LP relaxation: For each set $S \subseteq V$, and assignment $\alpha \in [P]^S$, we introduce an LP variable $y(S, \alpha)$ which is supposed to be an indicator that p agrees with α on S, i.e. $y(S, \alpha) = 1$ if the solution function p assigns the value $p(v) = \alpha(v)$ for all $v \in S$. For any two assignments $\alpha \in [P]^X$ and $\beta \in [P]^Y$ such that X and Y are disjoint, we write $\alpha \cup \beta$ to represent the assignment $\gamma \in [P]^{X \cup Y}$ that agrees with α on X and with β on Y. We use the following LP relaxation.

$$
\begin{aligned}
\max \quad & \sum_{e=(u,v) \in E} \sum_{\substack{\alpha \in [P]^{\{u,v\}}: \\ \alpha(u)+\alpha(v) \leq B_e}} (\alpha(u) + \alpha(v)) y(\{u,v\}, \alpha) \\
\text{s.t.} \quad & \sum_{\beta \in [P]^T} y(S \cup T, \alpha \cup \beta) = y(S, \alpha) \quad \forall S, T \subseteq V : |S \cup T| \leq r, S \cap T = \emptyset, \alpha \in [P]^S \\
& 0 \leq y(S, \alpha) \leq 1 \qquad \forall S : |S| \leq r, \forall \alpha \in [P]^S \\
& y(\emptyset, \emptyset) = 1
\end{aligned}
\tag{LP-r}
$$

The size of the LP is $P^{O(r)}$. A natural way to view this LP is to treat the variables $\{y(S, \alpha)\}_\alpha$ for a fixed set $S \subseteq V$ as a probability distribution where the value of $y(S, \alpha)$ represents the

probability that the vertices in set S are assigned price α. Notice that we have the constraint $1 = y(\emptyset, \emptyset) = \sum_{\alpha \in [P]^S} y(S, \alpha)$.

We first argue that this LP is indeed a relaxation for GVP. Let p^* be an optimal (integral) price function. For any set $S \subseteq V$ and any assignment α for S, we assign $y(S, \alpha) = 1$ if and only if $p^*(v) = \alpha(v)$ for all $v \in S$. Consider any two subsets $S, T : S \cap T = \emptyset$ and any assignment $\alpha \in [P]^S$, and notice that $y(S, \alpha) = 1$ if and only if there exist $\beta \in [P]^T$ such that $y(S \cup T, \alpha \cup \beta) = 1$. Therefore, the above solution satisfies all constraints, and the objective value equals to the total revenue collected by p^*.

Now let G be an input graph with treewidth k. First, we solve (LP-$2k$) and denote the optimal objective value by OPT. We describe below a randomized algorithm that returns price function p with expected total profit of OPT. This algorithm can be derandomized by the method of conditional expectation.

Algorithm description: Denote by $(T, \mathcal{V})$ the tree decomposition of graph G. Initially we have price function p where $p(v)$ is undefined for all $v \in V$. We process each element of the tree $t \in T$ in order defined by the tree T from root to leaves, ensuring that whenever any node $t \in T$ is processed, all ancestors of t have already been processed. When the algorithm processes the node $t \in T$, it assigns the prices to vertices in V_t, defining the values $p(v)$ for all $v \in V_t$. It is clear that function p will be defined for all $v \in V$ in the end.

Let $t \in T$ be the current tree node that is being processed. If t is the root of the tree, we assign the prices to vertices in V_t according to the probability distribution $\{y(V_t, \alpha)\}_{\alpha \in [P]^{V_t}}$. Otherwise, let t' be the parent of t. Suppose $\beta \in [P]^{V_{t'}}$ is a price assignment of vertices $V_{t'}$. Define $X' = V_t \setminus V_{t'}$ to be the set of vertices in V_t whose prices have not been assigned. We pick a random assignment $\alpha \in [P]^{X'}$ using the distribution $y(V_{t'} \cup X', \beta \cup \alpha)/y(V_{t'}, \beta)$, i.e. for each assignment $\alpha' \in [P]^{X'}$, $\mathbf{Pr}[\alpha = \alpha'] = y(V_{t'} \cup X', \beta \cup \alpha')/y(V_{t'}, \beta)$. Notice that this is a valid probability distribution because of the constraint

$$\sum_{\alpha' \in [P]^{X'}} y(V_{t'} \cup X', \beta \cup \alpha') = y(V_{t'}, \beta)$$

In the end of the algorithm, the following property holds.

Lemma 5.1. For any $t \in T$, the price $p\,|_{V_t}$ has the same distribution as $\{y(V_t, \alpha)\}_{\alpha \in [P]^{V_t}}$. In other words, for each tree node $t \in T$, we have

$$(\forall \alpha \in [P]^{V_t})\ \mathbf{Pr}[p|_{V_t} \text{ agrees with } \alpha] = y(V_t, \alpha)$$

Proof. We prove by induction on the ordering of tree nodes by tree T. For the root of the tree, the probability that $p|_{V_t}$ equals α is exactly $y(V_t, \alpha) = y(V_t, \alpha)$. Now we consider any non-root tree node t, and assume that the lemma holds for the parent tree node $t' \in T$. For any assignment

$\alpha \in [P]^{V_t}$, we have

$$\begin{aligned}
\mathbf{Pr}[p\,|_{V_t} \text{ agrees with } \alpha] &= \sum_{\substack{\beta \in [P]^{V_{t'}}: \\ \beta|_{V_t \cap V_{t'}} = \alpha|_{V_t \cap V_{t'}}}} \mathbf{Pr}[p\,|_{V_{t'}} = \beta]\mathbf{Pr}[p \text{ agrees with } \alpha \mid p|_{V_{t'}} = \beta] \\
&= \sum_{\beta : \beta|_{V_t \cap V_{t'}} = \alpha|_{V_t \cap V_{t'}}} y(V_{t'}, \beta) \frac{y(V_t \cup V_{t'}, \alpha \cup \beta)}{y(V_{t'}, \beta)} \\
&= \sum_{\beta : \beta|_{V_t \cap V_{t'}} = \alpha|_{V_t \cap V_{t'}}} y(V_t \cup V_{t'}, \alpha \cup \beta) \\
&= \sum_{\tilde{\beta} \in [P]^{V_{t'} \setminus V_t}} y(V_t \cup (V_{t'} \setminus V_t), \alpha \cup \tilde{\beta}) \\
&= y(V_t, \alpha)
\end{aligned}$$

Note that the second line follows from the first line by the induction hypothesis. The rest is simply a calculation. □

Now assuming the lemma, we argue that the expected profit collected by the algorithm is at least OPT: For each customer edge $uv \in E$, the expected profit made by uv equals to

$$\sum_{\alpha : \alpha(u) + \alpha(v) \le B_e} (\alpha(u) + \alpha(v)) \mathbf{Pr}[p \text{ agrees with } \alpha]$$

Since both end vertices of each edge $e = (u, v)$ belong to some set V_t, from the Lemma 5.1, the distribution of $p|_{\{u,v\}}$ is the same as that of $\{y(\{u, v\}, \alpha)\}_{\alpha \in [P]^{\{u,v\}}}$. So the probability term can be replaced by $y(\{u, v\}, \alpha)$.

Bounded genus graphs: One can show that the integrality gap of (LP-r) is at most $1 + \epsilon$ for $r = O(g/\epsilon)$. The proof follows along the same line as in Section 4.2, but we work with LP solution instead. We sketch it here for completeness. Suppose we have optimal LP solution $\{y(S, \alpha)\}_{S,\alpha}$, and let OPT_{LP} denote the LP-cost of this solution. Let $K = 1/\epsilon$. We create K sub-instances $G_1, \ldots, G_K$ by the same method as in Section 4. Let $r(E_i)$ be the total LP-cost collected from edges in E_i. Notice that instance G_i has total LP-cost of $\mathsf{OPT}_{LP} - r(E_i)$, so there must be instance G_{i^*} with total LP-cost of $(1 - \epsilon)\mathsf{OPT}_{LP}$. By solving sub-instance G_{i^*} using the LP rounding algorithm for bounded treewidth instances, we will get a pricing that collects revenue of $(1 - \epsilon)\mathsf{OPT}_{LP}$ (in Section 4, we need to solve all sub-instances and return the one with maximum revenue). So the integrality gap is at most $1/(1 - \epsilon) \le (1 + 2\epsilon)$.

6 k-partite graphs and bounded-degree graphs

We note that, by using the same arguments as in Theorem 4.1, one can show that GVP is APX-hard even on graphs of constant degrees and k-partite graphs when k is constant.

6.1 Improved approximation algorithms on k-partite graphs

Let G be a k-partite graph. We assume that we know the partition G into k partitions. To avoid confusion, we sometimes called each partition a *color class*. First we randomly partition the color classes into two roughly equal sides called L and R, i.e. we ensure that the number of color classes in L differ by those in R by at most one. We say that an edge e is cut if exactly one endpoint of e belongs to L. We analyze this algorithm in two cases using counting arguments. For the sake of analysis, we think of each node as having a unique integer ID.

k is even: Notice that the number of possible cuts is $\binom{k-1}{k/2-1}$. Each edge e belongs to exactly $\binom{k-2}{k/2-1}$ cuts (after fixing two end vertices of the edge, we pick $k/2-1$ more vertices to join the side of the node with smaller ID). Therefore, the probability that each edge is cut is

$$\frac{\binom{k-2}{k/2-1}}{\binom{k-1}{k/2-1}} = k/2(k-1)\,.$$

Observe that this term is slightly more than $1/2$. Using the algorithm of Balcan and Blum [BB07] on bipartite graphs, we get an approximation ratio of $4(k-1)/k$.

k is odd: In this case, the number of possible cuts is $\binom{k}{\lfloor k/2 \rfloor}$. Each edge will be in $2\binom{k-2}{\lfloor k/2 \rfloor -1}$ cuts (after fixing two end vertices of the edge, we pick $\lfloor k/2 \rfloor - 1$ more vertices to join one side out of two sides). It is easy to see that this term is equal to $\binom{k-2}{\lfloor k/2 \rfloor -1} + \binom{k-2}{\lfloor k/2 \rfloor} = \binom{k-1}{\lfloor k/2 \rfloor}$. Hence the probability that each edge is cut is $\frac{\binom{k-1}{\lfloor k/2 \rfloor}}{\binom{k}{\lfloor k/2 \rfloor}} = \frac{k+1}{2k}$. Again, using the algorithm of Balcan and Blum [BB07] on bipartite graphs, we get an approximation ratio of $\frac{4k}{k+1}$.

In general graphs, we may think that $k = n$, so this will give a slightly improved approximation factor of $4(1-1/n)$.

Deterministic algorithm: We use the method of conditional expectation. Put an arbitrary color class on the left side L. For each color class, determine whether we should put it in L or R as follows. Put the color class on the left side and calculate the conditional expectation (which can be computed since we can compute a probability that an edge is in the cut which depends on the number of vertices already present in L and R). Try putting the color class in R and do the same. Pick the choice that gives better expected revenue.

Tightness of the algorithm: Our approximation factor here is, in some sense, tight: We argue that improving this bound further for any even number k would immediately imply an improved approximation ratio for general graphs. Suppose we have an α approximation algorithm for k-partite graphs where $\alpha < 4(k-1)/k$. We randomly partition nodes into k parts and delete edges that connect two vertices in the same set of the partition. Each edge is deleted with probability $1/k$. Then we run the α-approximation algorithm for the resulting k-partite graph, and this would give us an approximation ratio of $\alpha k/(k-1) < 4$ for general graphs.

6.2 Polynomial-time algorithms for Graphs of degree at most two

We observe that when the input graph has degree at most two, i.e. when in consists of paths and cycle, the problem can be solved in polynomial time. This can be done by considering two cases: when every edge contributes a non-zero revenue to the optimal solution and otherwise. In the former case, we can solve the problem by writing a linear program. In the latter case, the problem is solved by a dynamic programming technique.

We observe that for any graph $G(V, E)$ if every edge $e \in E$ contributes positive amount to the total revenue of the optimal solution then the optimal vertex prices can be calculated using the following linear program :

$$
\begin{aligned}
(LP_{opt}) \qquad & \\
\max \quad & \sum_{e=uv \in E} (p(u) + p(v)) \\
\text{s.t.} \quad & p(u) + p(v) \leq B_e \\
& p(u) \geq 0 \text{ for all } u \in V
\end{aligned}
$$

Let $LP_{opt}(G)$ represent the optimal value of LP_{opt} on a graph G. Let $P = \{v_1, v_2, \ldots, v_n\}$ be a path with edges (v_i, v_{i+1}) for $1 \leq i \leq n-1$. Let $OPT(G)$ represent the optimal revenue that can be obtained from the graph G. For $i < j$, let $R[v_i, v_j]$ be the maximum revenue that can be obtained from the subpath (say P_{ij}) induced by the vertices $v_i, v_{i+1}, \ldots, v_j$. Note that $R[v_i, v_i] = 0$ for all $1 \leq i \leq n$. For $i < j$, $R[i, j]$ can be computed using the following recursion.

$$
R[i,j] = \max \begin{cases} LP_{opt}(P_{ij}) & \text{if all the edges of } P_{ij} \\ & \text{contribute positive revenue.} \\ \max_{k=i,\ldots,j-1} R[v_i, v_k] + R[v_{k+1}, v_j] & \text{otherwise} \end{cases}
$$

$OPT(P) = R[1, n]$ is the optimal revenue obtained from the path P. By solving the above recursion we obtain a polynomial time algorithm for paths.

Now we design a polynomial time algorithm for cycles. Let $C = v_1, v_2, \ldots, v_n$ be a cycle with edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1)$. If all the edges of C contribute positive revenue then the maximum revenue is given by $LP_{opt}(C)$. If one of the edges $uv \in C$ contributes zero to the total revenue then the maximum revenue is obtained by using the above mentioned algorithm on the path obtained by deleting uv from C. Hence we get the following recursion.

$$
OPT(C) = \max \begin{cases} LP_{opt}(C) & \text{if all the edges of } C \\ & \text{contribute positive revenue} \\ \max_{e \in V(C)} OPT(C \setminus e) & \text{otherwise} \end{cases}
$$

Since $C \setminus e$ is a path, we use the previous algorithm to compute $OPT(C \setminus e)$ in polynomial time. Solving this recursion we get a polynomial time algorithm for cycles.

Any graph of degree at most two consists of paths and cycles. By combining the above mentioned algorithms for paths and cycles we get a polynomial time algorithm for graphs of degree at most two.

6.3 Graphs of degree at most four

By using the algorithm mentioned in Section 6.1, we get factor 3 approximation algorithms on graphs with degree at most three and four. Now we present a better algorithm achieving an approximation factor of 2 for graphs of degree at most four.

Let $G = (V, E)$ be a graph with degree at most four. We decompose edges of G into $E = E_1 \cup E_2$ such that the corresponding subgraphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ have degree at most two: This is a standard trick. First add an arbitrary matching M between odd-degree vertices, so the resulting graph is Eulerian. Let C be its Eulerian tour. Then we let $E_1 = C \setminus M$ and $E_2 = E \setminus E_1$.

Using the algorithm from Section 6.2 we can compute optimal solutions to G_1 and G_2 in polynomial time. Note that $\mathsf{OPT}(G_1) + \mathsf{OPT}(G_2) \geq \mathsf{OPT}(G)$, so we get 2-approximation.

7 Open Problems

We settled the complexity of GVP on graphs of bounded genus. Since the integrality gap of Sherali-Adams relaxation for these cases is $(1+\epsilon)$, it is interesting to see how this integrality gap behaves in general graphs. Using lift-and-project LP might be a possible way to get improved approximation algorithms or better hardness results. Note that, for two rounds of Sherali-Adams relaxation, the integrality gap is at least $4 - \epsilon$ by Khandekar et al. We also considered the bounded degree cases and presented a 2-approximation algorithm for cubic graphs. It is an interesting open problem to improve this approximation factor. We believe that GVP remains NP-hard even on trees.

References

[AGG07] Isolde Adler, Georg Gottlob, and Martin Grohe. Hypertree width and related hypergraph invariants. *Eur. J. Comb.*, 28(8):2167–2181, 2007. 13

[Bak94] Brenda S. Baker. Approximation algorithms for np-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994. 2

[BB07] Maria-Florina Balcan and Avrim Blum. Approximation algorithms and online mechanisms for item pricing. *Theory of Computing*, 3(1):179–195, 2007. Also in EC'06. 1, 2, 9

[BÖ04] Daniel Bienstock and Nuri Özbay. Tree-width and the sherali-adams operator. *Discrete Optimization*, 1(1):13–21, 2004. 2

[Bod96] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996. Also in STOC'93. 2, 3, 13

[Bro41] R. L. Brooks. On colouring the nodes of a network. *Proc. Cambridge Philosophical Society, Math. Phys. Sci.*, 37:194–197, 1941. 2, 3

[CDF+09] Jean Cardinal, Erik D. Demaine, Samuel Fiorini, Gwenaël Joret, Ilan Newman, and Oren Weimann. The stackelberg minimum spanning tree game on planar and bounded-treewidth graphs. In *WINE*, pages 125–136, 2009. 2

[CKR10] Eden Chlamtac, Robert Krauthgamer, and Prasad Raghavendra. Approximating sparsest cut in graphs of bounded treewidth. In *APPROX-RANDOM*, pages 124–137, 2010. 2

[DHiK05] Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *FOCS*, pages 637–646, 2005. 5

[Epp00] David Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3):275–291, 2000. Also in SODA'95. 2

[GGM+05] Georg Gottlob, Martin Grohe, Nysret Musliu, Marko Samer, and Francesco Scarcello. Hypertree decompositions: Structure, algorithms, and applications. In *WG*, pages 1–15, 2005. 13

[GHK+05] Venkatesan Guruswami, Jason D. Hartline, Anna R. Karlin, David Kempe, Claire Kenyon, and Frank McSherry. On profit-maximizing envy-free pricing. In *SODA*, pages 1164–1173, 2005. 1

[GJ77] M. R. Garey and David S. Johnson. The rectilinear steiner tree problem in np complete. *SIAM Journal of Applied Mathematics*, 32:826–834, 1977. 4

[Hea90] P. J. Heawood. Map colour theorem. *Quarterly Journal of Pure and Applied Mathematics*, 24:332–338, 1890. 2

[KKMS09] Rohit Khandekar, Tracy Kimbrel, Konstantin Makarychev, and Maxim Sviridenko. On hardness of pricing items for single-minded bidders. In *APPROX-RANDOM*, pages 202–216, 2009. 1, 2, 3, 14

[KMN10] Anna R. Karlin, Claire Mathieu, and C. Thach Nguyen. Integrality gaps of linear and semi-definite programming relaxations for knapsack. *CoRR*, abs/1007.1283, 2010. 2

[KMR11] Robert Krauthgamer, Aranyak Mehta, and Atri Rudra. Pricing commodities. *Theor. Comput. Sci.*, 412(7):602–613, 2011. Also in WAOA'07. 1

[RS84] Neil Robertson and Paul D. Seymour. Graph minors. iii. planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984. 3, 4, 13

[SA90] Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. Discrete Math.*, 3(3):411–430, 1990. 2

Appendix

A Proof of Lemma 2.1

Let $B_{\max}$ denote the maximum budget among all consumers, and $M = \epsilon B_{\max}/nm$. Notice that $B_{\max} \leq \mathsf{OPT}$. We create a new instance of the problem as follows: For each consumer $uv \in E$, we define new budget $B'_e = \lfloor B_e/M \rfloor$. Let p^* be the optimal price for the old instance. We define the price $p'(v) = \lfloor p^*(v)/M \rfloor$.

First it is clear that the first property holds because $B_e/M \leq O(mn/\epsilon)$. To prove the second property, let OPT' denote the optimal revenue of the new instance. Clearly, $M\mathsf{OPT}' \leq \mathsf{OPT}$, since any price of new instance can be turned into price of the old instance that collects M times as much. Next, for any consumer $e = uv$ that collects revenue $p^*(u) + p^*(v)$ in the old instance, we have $p'(u)+p'(v) \leq \lfloor (p^*(u) + p^*(v))/M \rfloor \leq B'_e$, so this consumer does not go over budget with price p' in the new instance. And $\sum_{e\in E} Mr'(e) \geq \sum_{e\in E} (r(e) - 2M) \geq (1-\epsilon)\mathsf{OPT} \geq (1-\epsilon)M\mathsf{OPT}'$. Therefore, the revenue collected by p' is at least $(1-\epsilon)\mathsf{OPT}'$.

Observe that any α-approximation algorithm for the new instance can be turned into $(1+2\epsilon)\alpha$ approximation algorithm for the old instance as follows: Let p' be the price that collects a total revenue of OPT'/α in the new instance. We define $p(v) = Mp'(v)$ for all $v \in V$. Then the total revenue we get is $\frac{M\mathsf{OPT}'}{\alpha} \geq \frac{(1-\epsilon)\mathsf{OPT}}{\alpha} \geq \frac{\mathsf{OPT}}{(1+2\epsilon)\alpha}$.

B Extension to bounded tree-width hypergraphs

We note that the FPTAS can be extended to hypergraphs of bounded tree-width defined naturally as follows. Consider a hypergraph H of width k (following the definition in [RS84]). This means that there is a *tree-decomposition* $(T, \mathcal{V})$ of H of width k; that is, there is a tree T and $\mathcal{V} = (V_t)_{t\in T}$, a family of vertex sets $V_t \subseteq V(H)$ indexed by the vertices t of T, with the following properties.

1. $V(H) = \bigcup_{t\in T} V_t$;
2. for every edge $e \in E(H)$ there exists $t \in T$ such that $e \subseteq V_t$;
3. for any $v \in V(H)$, if $v \in V_{t_1}$ and $v \in V_{t_2}$ then $v \in V_{t_3}$ for any t_3 in the (unique) path between t_1 and t_2;

It is observed that if the tree-width of H is bounded by a constant, then its tree decomposition can be constructed, as follows. For any hypergraph H, we construct a graph G, called a *primal* graph of H by letting

$$G = (V(H), \{(u, v) \mid u \neq v, \text{there exists } e \in E(H) \text{ such that } u, v \in e\}) .$$

It can be observed (see, e.g., [GGM+05, AGG07]) that $(T, \mathcal{V})$ is a tree decomposition of H if and only if it is a tree decomposition of G. Note again that we can recognize if the tree-width of G is a fixed constant and, if it is, construct a tree decomposition of G in linear time [Bod96]. Assuming that we have the tree decomposition $(T, \mathcal{V})$, we solve the problem in time $\text{poly}(|T|, P^k)$, where P is the maximum possible budget from Lemma 2.1, in the same way as in Section 3 except that we have to define E_t for hyperedges instead of edges. This is done by defining $t(e)$ for each hyperedge e to be the node t in tree T such that, among $t' \in T$ such that $V_{t'} \subseteq e$, t is nearest to the root.

Note that such node exists since, for any clique C, there is a node t such that V_t contains all nodes in C. The rest of the algorithm remains the same.

C Generating (LP-r) from Lift-and-project operations

In this section we show that the constraints in (LP-r) can be automatically generated by applying r rounds of Sherali-Adams lift-and-projects on the base LP. There are many ways to write a natural LP relaxation for this problem, but one natural choice is the following: For each vertex v and each possible price $i \in [P]$, we introduce variable $x(v,i)$, whose supposed value is to indicate that the price of v is i. For each pair of vertices $u, v \in V$, we have variable $z(u,v,i,j)$ to be an indicator of $p(u) = i$ and $p(v) = j$.

(LP')

$$\begin{aligned}
\max \quad & \sum_{(u,v)\in E} \sum_{i+j\le B_e} (i+j) z(u,v,i,j) \\
\text{s.t.} \quad & \sum_{i\in[P]} x(v,i) = 1 \qquad \text{for all } v \in V \\
& \sum_{i,j\in[P]} z(u,v,i,j) = 1 \qquad \text{for all } u,v \in V \\
& z(u,v,i,j) \ge x(u,i) + x(v,j) - 1 \qquad \text{for all } u,v \in V,\ i,j \in [P] \\
& x(u,i), z(u,v,i,j) \in [0,1]
\end{aligned}$$

The last set of constraints enforces (in the integral world) that if $x(u,i) = 1$ and $x(v,j) = 1$, then $z(u,v,i,j)$ must be set to 1. It is easy to see that this LP has bad integrality gap . We remark that the LP relaxation used by Khandekar et al. [KKMS09] is equivalent to (LP') after two rounds of sherali-adams lift-and-project operations, and has integrality gap upper bound of 4 in general graphs.

Let K_r be the polytope of feasible solution to (LP-r) and K' be the feasible polytope for (LP'). Denote by $\mathsf{SA}^r(K')$ the polytope obtained after applying r rounds of lift-and-projects to K'. We will not derive all the constraints that define $\mathsf{SA}^r(K')$, but instead we will only show that $\mathsf{SA}^r(K') \subseteq K_r$. This is sufficient for us to conclude that $O(k)$ rounds of Sherali-Adams relaxation are enough to solve the graph pricing problem where the graph has tree-width at most k.

Lemma C.1. For all $r \ge 2$, $\mathsf{SA}^r(K') \subseteq K_r$

Proof. We prove by induction on r that constraints of (LP-r) can be generated by r rounds of lift operations. For $r = 2$, we show how to derive all the constraints of (LP-2) using at most 2 rounds. We introduce variables $y'(S,\alpha)$ for each set $S \subseteq V, |S| \le 2$ and for each assignment $\alpha \in [P]^S$, such that $y'(\{u\}, i) = x(u,i)$ for all $u \in V$ and $i \in [P]$. We let $y'(\emptyset,\emptyset) = 1$. It is easy to see that variables $y'(S,\alpha)$ correspond exactly to the variables $y(S,\alpha)$ in (LP-2), and all constraints can be generated. The only nontrivial part is to show that, after two rounds of lifts, the constraint $y'(\{u,v\},\{i,j\}) = z(u,v,i,j)$ holds for all $u,v \in V$ and $i,j \in [P]$. We sketch a proof of this fact here: First, apply $y'(u,i)$ to the third set of constraints to get

$$y'(u,i) \star z(u,v,i,j) \ge y'(\{u,v\},\{i,j\})$$

By applying $z(u,v,i,j)$ to the inequality $y'(u,i) \leq y(\emptyset,\emptyset)$, we can get $y'(u,i) \star z(u,v,i,j) \leq z(u,v,i,j)$, so this implies $y'(\{u,v\},\{i,j\}) \leq z(u,v,i,j)$. By summing over i,j, we get $1 = \sum_{i,j\in[P]} y'(\{u,v\},\{i,j\}) \leq \sum_{i,j\in[P]} z(u,v,i,j) = 1$, so the inequality has to be tight for all i,j, i.e. $z(u,v,i,j) = y'(\{u,v\},\{i,j\})$, as desired.

Next we show that, by applying one round of lift-and-project to (LP-r), we get all the constraints in (LP-$(r+1)$). It suffices to consider only the constraints that involve $S,T \subseteq V$ such that $|S \cup T| = r+1$. Let S be a non-empty set (if it was empty, the claim follows trivially), so we can write $S = \{v\} \cup S'$ and $\alpha' = \alpha|_{S'}$. Notice that constraint $\sum_{\beta\in[P]^T} y(S' \cup T, \alpha' \cup \beta) = y(S',\alpha')$ belongs to (LP-r). By applying Sherali-Adams operation $\{y(\{v\},\alpha(v))\}\star$ to both sides, we get the desired inequality. □

Corollary C.2. If the input graph has tree-width at most k, then the Sherali-Adams polytope $\mathsf{SA}^{2k}(K')$ has integrality gap exactly one.